

Python! Raw LiDAR XYZI to First/Last Return Grid

Contributed by Kevin Bell
 23, Mar. 2009
 Last Updated 23, Mar. 2009

Have you ever needed to convert raw LiDAR xyz files to first/last return grids? Me too, so I hacked up a routine to git 'er done. I'll write out the idea below and see if you can follow along in the code.

- make a list of all the xyz files in a directory
- the points are dense, but random, so eliminate the decimals which gives you concrete x y pairs
- don't drop the z decimals
- concatenate the x and y
- for each xy pair, make a list of z's (we can have multiple laser strikes per meter)
- sort each xy pair list
- write out the new x,y,z's to a csv file, but keep only the highest or lowest z
- make an event layer
- point to grid
- FocalMean to fill in the wholes (i'm making a 1 meter grid from 1.25 meter postings)
- delete your temp data.

To follow this, you'll need to understand python "lists", "dictionaries", "slicing", and built in commands. Realize that the paths are hard coded, although its only a few lines.

```
#xyz_to_firstAndLastReturn.py

#Kevin Bell

#20090322

import csv
import os
import time
import arcgisscripting

gp = arcgisscripting.create(9.3)
gp.workspace = r"F:\gis\zMisc\20090318_buildFirstReturnFromXYZ"
gp.overwriteoutput = 1

try:
    gp.CheckOutExtension("spatial")
    print "acquired spatial analyst license"
except:
    print "no spatial analyst license available... bailing out!"
    os.sys.exit()

startTime = time.clock()

d = {}

def __buildDict(x, y, z):
    if d.has_key(x + '|' + y):
        d[x + '|' + y].append(z)
    else:
        d[x + '|' + y] = []
        d[x + '|' + y].append(z)
```

```

def __convertRawXYZtoReturnCSV(f, firstLast): # NO XYZ header in original xyzi file!
    # f = 12TVL080080.xyz
    print "beginning to read file"
    theReader = csv.reader(open(f))
    if firstLast == 'last':
        writer = csv.writer(open('last_' + f[:-4] + ".csv", "wb")); print "made the file"
    writer = csv.writer(open('first_' + f[:-4] + ".csv", "wb")); print "made the file"

    i = 0
    for r in theReader:
        i += 1
        if i == 1:
            header = "XYZ"
            writer.writerow(header)
        if str(i).endswith("00000"):
            print i
        #print r #[408748.990, '4508087.200', '1290.760', '182']
        rr = r[0][:-4] + " | " + r[1][:-4], r[2], r[3] #('408748 | 4508087', '1290.760', '182')
        x = r[0][:-4]
        y = r[1][:-4]
        z = r[2]
        __buildDict(x,y,z)
        del x,y,z

    print "...beginning to write extracted returns to file"
    for k in d.iterkeys():
        x,y=k.split("|")
        d[k].sort()
        if firstLast == 'last':
            d[k].reverse() # use for last return
        z = d[k].pop()
        rr = x, y, z
        writer.writerow(rr)

    print "finished convert raw xyzi to csv"

def convertReturnCSVtoGrid(infile, firstLast): # now it has the XYZ header
    __convertRawXYZtoReturnCSV(infile, firstLast) ## is this right?

    gp.MakeXYEventLayer_management ('first_' + infile[:-4] + ".csv", 'X', 'Y', r'Temp\subsetLyr')
    print "made XY layer"

    gp.PointToRaster_conversion (r'Temp\subsetLyr', 'Z', r'Temp\rasSubset', 'MOST_FREQUENT', 'NONE', '1')
    print "completed Point to Raster"

    #outFilledRaster = r'Temp\outFilledRas' ## must change output name
    if firstLast == 'last':
        outFilledRaster = r'Temp' + '\\' + 'las' + infile[5:11]
    else:
        outFilledRaster = r'Temp' + '\\' + 'fir' + infile[5:11]

    InExpression = 'FocalMean(Temp\rasSubset, Circle, 1)'
    gp.SingleOutputMapAlgebra_sa(InExpression, outFilledRaster)
    print "completed MapAlgebra"

    gp.delete(r'Temp\subsetLyr'); print "deleted the temp lyr"
    gp.delete(r'Temp\rasSubset'); print "deleted the temp raster"
    d.clear()
    print "/n"

#""oooooooooooooooooooooooooooooooooooooooooooooooooooo"""

filelist = os.listdir(r"F:\gis\zMisc\20090318_buildFirstReturnFromXYZ\OriginalData")

```

```
filelist = [x for x in filelist if x.endswith(".xyz")]
```

```
for f in filelist:
```

```
    convertReturnCSVtoGrid(f, 'first')
    convertReturnCSVtoGrid(f, 'last')
```

```
stopTime = time.clock()
elapsedTime = stopTime - startTime
elapsedTime = elapsedTime / 60
print "Time for operation:"
print round(elapsedTime, 1)
print "minutes"
```